1)Qual a saída do algoritmo?
Irá mostrar páginas de 4 colunas e 50 linhas com números primos até que K = 1000, resultando no número final: 7919.

2)Você julga que este código é limpo? Aponte quais erros o programador cometeu que prejudicaram a qualidade do código. Obs: não existe nenhum bug escondido no código.
Não.
Nomeação das variáveis - Não foi utilizadas variáveis com nomes descritivos, apenas 'J', 'K', 'N'. Não foi utilizado camelCase para nomeação, apenas variáveis com letras maiusculas, como: 'PAGENUMBER', 'ORDMAX', 'ROWOFFSET'.
Duplicação do código - Duplicação das variáveis com alteração dos números.

4) Explique como o conceito de middlewares no Express.js pode ser utilizado para evitar repetição de código.

Pode ser utilizado de várias formas e para vários casos. Por exemplo: caso haja um código que deve ser utilizado em várias rotas ou controladores, o middleware pode guardar esse código e ser utilizado durante essas requisições.
Pode também ser usado para validação de dados, autenticação e autorização do usuário, por exemplo quando temos dados de entrada nas requisições. Ao invés de repetir as validações em todas as rotas, podemos criar um middleware que execute essa função.

5)Tendo em vista duas abordagens de backend: uma utilizando um ORM (como Prisma e Sequelize) e outra utilizando apenas um query builder (como o Knex), quais as vantagens e desvantagens de cada abordagem?

Utilizando um ORM temos como vantagens abstração do banco de dados, transformando as tabelas em objetos e propriedades, facilitando a legibilidade do código. Aumento da produtividade pois reduz a quantidade de código. Traz mais segurança e ajuda a evitar vulnerabilidade e pode gerenciar automaticamente relacionamentos entre tabelas, evitando ter de ser feito manualmente.
Entre as desvantagens temos em alguns casos onde o ORM pode atrapalhar o desempenho sistema, pois o mesmo adiciona uma camada extra. Outra desvantagem são suas restrições que podem limitar a forma como desenvolvedores trabalham com o banco de dados.
Ao se utilizar um query builder temos como vantagens as limpezas dos inputs, onde os métodos e as funções ficam responsáveis pela limpeza dos inputs do usuário. Outra vantagem é a facilidade na escrita.
Entre as desvantagens há a pouca abstração que requer uma boa compreensão de como utilizar o SQL por parte do programador. Outra desvantagem é a limitação complexa, podendo ultrapassar a capacidade do query, precisando assim recorrer ao SQL tradicional.

6)

```
-- Criar tabela jogador
CREATE TABLE jogador (
   id NUMBER,
   nome VARCHAR2(100),
   idade NUMBER
);
```

```
-- Criar tabela partidas
CREATE TABLE partidas (
   id NUMBER,
   jogador1_id NUMBER,
   jogador2_id NUMBER,
   pontos_jogador1 NUMBER,
   pontos_jogador2 NUMBER,
   duracao INTERVAL DAY TO SECOND
);

-- Inserir jogadores
INSERT INTO jogador (id, nome, idade)
VALUES (1, 'Jogador1', 25);

INSERT INTO jogador (id, nome, idade)
VALUES (2, 'Jogador2', 28);

-- Inserir partidas
INSERT INTO partidas (id, jogador1_id, jogador2_id, pontos_jogador1, pontos_jogador2, duracao)
VALUES (1, 1, 2, 15, 20, INTERVAL '2' HOUR);

INSERT INTO partidas (id, jogador1_id, jogador2_id, pontos_jogador1, pontos_jogador2, duracao)
VALUES (2, 2, 1, 25, 10, INTERVAL '1' HOUR);

INSERT INTO partidas (id, jogador1_id, jogador2_id, pontos_jogador1, pontos_jogador2, duracao)
VALUES (3, 1, 2, 10, 25, INTERVAL '2' HOUR);

INSERT INTO partidas (id, jogador1_id, jogador2_id, pontos_jogador1, pontos_jogador2, duracao)
VALUES (4, 1, 2, 20, 15, INTERVAL '2' HOUR);

INSERT INTO partidas (id, jogador1_id, jogador2_id, pontos_jogador1, pontos_jogador2, duracao)
VALUES (5, 2, 1, 10, 20, INTERVAL '1' HOUR);

--Consulta
SELECT DISTINCT j1.nome AS jogador1, j2.nome AS jogador2
FROM jogador j1
JOIN partidas p ON j1.id = p.jogador1_id
JOIN jogador j2 ON j2.id = p.jogador2_id
WHERE p.pontos_jogador1 + p.pontos_jogador2 > 30
   AND p.duracao > INTERVAL '90' MINUTE
GROUP BY j1.nome, j2.nome
HAVING COUNT(*) > 2;
```